SPECIFICATION AMENDMENT(S)

In the Specification:

Please replace paragraph number [00012] with the following amended paragraph:

A1

[00012] Figure 4A is a diagram illustrating data flow through either of the deframing ~~slice 301~~ slices 303 or 305 of Figure 3 according to one embodiment of the invention.

Please replace paragraph number [00013] with the following amended paragraph:

A2

[00013] Figure 4B is a diagram illustrating data flow through ~~either of~~ the deframing ~~slices 303 or 305~~ slice 301 of Figure 3 according to one embodiment of the invention.

Please replace paragraph number [00031] with the following amended paragraph:

A3

[00031] Figure 18 is a diagram illustrating the organization of per-alignment state machines in the memory unit ~~321~~ 323 of Figure 3 according to one embodiment of the invention.

Please replace paragraph number [00037] with the following amended paragraph:

A4

[00037] Figure 2 is a diagram of units of a network element according to one embodiment of the invention. In Figure 2, a receiving unit 201 receives multiple DS3 signals. The DS3 signals can loop to a transmit buffering unit 242. The receiving unit 201 is also connected to an optical transmitting unit 225 and a receive buffering unit 202. The

receive buffering unit 202 is connected to a DS*n* deframing block 250. The optical transmitting unit ~~25~~ 225 processes the DS3 signals from the receiving unit 201 for optical transmission (e.g., mapping the DS3 signals to STS formatting).

A4

Please replace paragraph number [00045] with the following amended paragraph:

[00045] The multiplexer 313 multiplexes the input selected by the selecting units 311 and 312 before sending the multiplexed input into a DS3 deframer 320. Each deframer slice includes the DS3 deframer 320, a DS2 deframer 322, and a DS1 deframer 324.

A5

Please replace paragraph number [00046] with the following amended paragraph:

[00046] Each individual deframer processes successively lower bandwidth channels. Since each deframer handles two DS3 bit streams worth of data, though, each deframer actually processes approximately the same total number of bits. The DS3 deframer 320 handles two DS3 channels. The DS2 deframer 322 processes fourteen DS2 channels. The DS1 deframer 324 processes fifty-six DS1 channels. Input flows from the DS3 deframer 320 to the DS2 deframer 322, and then to the DS1 deframer 324. From the DS1 deframer 324 of each of the deframing slices 301, 303, and 305, bits flow into the DS1 data buffer 213 of Figure 2. The bits from each of the deframing slices 301, 303 and 305 are respectively stored in one of the corresponding buffers 325-327 for bit to byte conversion. Once the data is converted, it is multiplexed by the multiplexing unit 328 and transmitted to the protocol receive unit 215.

A6

Please replace paragraph number [00047] with the following amended paragraph:

[00047] In addition to the DS3 inputs 302, 304 and inputs 306, 308, the deframing slice 301 receives DS1 bit streams from the receiving T1 buffer 207 of Figure 2. The receiving T1 buffer 207 includes a set of buffers 335 to buffer individual DS1 signals. The buffered DS1 signals are multiplexed by a multiplexing unit 333 of the receiving T1 buffer 207. The multiplexer 333 passes the multiplexed DS1 signals to the deframing slice 301. When the deframing slice 301 receives DS1 bit streams, it multiplexes the DS1 bit streams with one of the deframed bit streams 302, 304, 316 or 318 of the deframing slice. These inputs are multiplexed at a multiplexer 315 before being sent to the DS1 deframer 324. The DS1 deframer 324 of each of the deframing slices 301, 303 and 305, is connected to a memory controller 321. The memory controller 321 handles read and write operations to an external memory unit 323. The external memory unit 323 stores states for sync hunting which is described later in relation to Figure 7, ~~1A-1B~~8A 8B, and ~~JA-JB~~11A-11B. The memory controller 321 serves the DS1 deframer 324 of each deframing slice 301, 303, 305 at the same time. In an example of six deframer slices, each receiving two DS3 bit streams, the memory controller 321 iterates through 168 (6 slices * 28 DS1 channels per DS3) channels of possible DS1 sync hunting. In ~~one~~ another embodiment of the invention, the order of iteration is subchannel 0-27 for the first DS3 input bit stream (channel 0) followed by subchannels 0-27 for the second DS3 input bit stream (channel 1). In one embodiment of the invention, the memory controller 321 serves all read requests before serving all write requests in the order previously described. Processing requests in this fashion holds read to write bus turnaround to a minimum of once per 168 bus cycles in one embodiment of the invention.

Please replace paragraph number [00048] with the following amended paragraph:

[00048] In another embodiment of the invention, every deframing slice 301, 303, 305 only receives <u>one</u> DS3 bit stream input. In another embodiment of the invention, each deframer slice receives one DS3 bit stream input and a set of DS1 bit streams. In another embodiment of the invention, each deframing slice receives inputs from two sets of DS1 bit streams. In another embodiment of the invention, a deframing slice can have *N* inputs, each of the *N* inputs independently configurable for either a DS3 input or a set of DS1 inputs.

Please replace paragraph number [00050] with the following amended paragraph:

[00050] Figure 4A is a diagram illustrating data flow through either of the deframing slices 303 or 305 of Figure 3 according to one embodiment of the invention. The deframing slice 303 of Figure 3 is used as an illustration for Figure 4A. In Figure ~~C~~<u>4A</u>, a data bit stream 401 (from the selecting unit 312), a data bit stream 402 (from the selecting unit 311), and a channel select signal 403 flow into the multiplexer 313 of Figure 3. The data bit streams 401 and 402 may include bits from the original DS3 signals and valid bits. From the multiplexer 313, a multiplexed data bit stream 405, a valid bit stream 407 and a channel bit stream 409 flow into the DS3 deframer 320 of Figure 3. From the DS3 deframer 320, a data bit stream 406 and a valid bit stream 408 flow into the DS2 deframer at 322. A subchannel bit stream 410 flows into the DS2 deframer 322 and a context memory 411. The context memory 411 includes a per-channel state memory and a sync hunt per-alignment memory for each pair of subchannels, which will be described herein. Information 404 from the context memory 411 flows into the DS2 deframer 322. Updates 444 are written back to the context memory 411. A data bit stream 412 and a

validity bit stream 414 flow from the DS2 deframer 322 into a DS1 deframer 324. The subchannel bit stream 416 flows from the DS2 deframer 322 to both the DS1 deframer 324 and a context memory 417. Information 419 from the context memory 417 flows into the DS1 deframer 324. Updates 432 are written back to the context memory 417. A data bit stream 418, a valid bit stream 420, and a subchannel bit stream 422 flow from the DS1 deframer 324 out of the deframing slice 303.

A9

Please replace paragraph number [00053] with the following amended paragraph:

[00053] The DS3 deframer 320 processes the streams 405, 407 and 409 and generates a data bit stream 406, a validity bit stream 408, and a subchannel bit stream 410 which flow into the DS2 deframer 322. The subchannel bit stream 410 also flows into a context memory 411. The context memory 411 includes a per-channel state memory and a sync hunt per-alignment memory for each pair of subchannels. The per-channel state memory and the sync hunt per-alignment memory for each deframer will be described later herein with references to Figures ~~1-M~~6-12. Information 404 from the context memory 411 flows into the DS2 deframer 322.

A10

Please replace paragraph number [00054] with the following amended paragraph:

[00054] The DS2 deframer 322 processes the streams 406, 408, 410 and the information 404 from the context memory 411 to generate a data bit stream 413, a validity bit

A11

stream 415, and a subchannel bit stream 417. The streams 413, 415, and 417 flow into the multiplexer 315. Updates ~~444are~~ 444 are written back to the context memory 411 from the DS2 deframer 322. Data bit streams also flow into the multiplexer ~~313~~ 315 from the receiving T1 buffer 207. A data bit stream 427, a validity bit stream 425, and a subchannel bit stream 424 flow into the multiplexer 315 from the receiving T1 buffer 207. The data bit stream 427 and the data bit stream 413 are multiplexed to generate a data bit stream 412. The validity bit streams 415 and 425 are multiplexed to generate a validity bit stream 414. The subchannel bit streams 417 and 424 are multiplexed to generate the bit stream 416. The streams 412, 414, 416 flow into the DS1 deframer 324. The subchannel bit stream 416 also flows into a context memory 419. Information 430 from the context memory flows into the DS1 deframer 324. The context memory 419 and the information 430 stored in the context memory 419 are described later.

A11

Please replace paragraph number [00056] with the following amended paragraph:

[00056] Figure 5 is a diagram of the DS3 deframer 320 of Figure 3 according to one embodiment of the invention. In Figure 5, the DS3 deframer 320 receives bit streams from a source external to the DS3 deframer 320. The two DS3 data bit streams 401 and 402 of Figure 4A feed into the multiplexing unit 313 of Figure 3. The channel select signal 403 also feeds into the multiplexing unit 313. The multiplexing unit 313 multiplexes the DS3 bit streams 401 and 402 to create the multiplexed DS3 data bit stream 405 that is fed into the DS3 deframer 320

A12

along with the valid bit stream 407 and the channel bit stream 409 of Figure ~~C~~4. A dashed line 515 indicates a first pipe stage. In the first pipe stage, a per-channel state memory 511 sends information to a sync hunt per-alignment memory 513. The per-channel state memory 511 also sends information to a register 521. Bits indicating the per-alignment state are transmitted from the sync hunt per-alignment memory 513 to a register 523. Also in the first pipe stage, the data bit stream 409 is stored in a register 517 while the streams 405, 407 are stored in a register 519. A dashed line 533 indicates a second pipe stage of the DS3 deframer 320. In the second pipe stage, bits from the registers 517, 519, 521 and 523 flow to a DS3 deframing logic 525 and a DS3 sync hunt logic 527. The bits flowing from the register 517 indicate side information (i.e., channel). In this example, the side information from the register 517 indicates whether the bit stream from the register 519 is the DS3 bit stream 401 or the DS3 bit stream 402. Data from the register 521 indicates a global state for the DS3 deframer and a counter value indicating location within a subframe for a given DS3 signal. The global state is described later in more detail with reference to Figures 8A-8B. The bits from the register 523 indicate the per-alignment state. Output from the DS3 sync hunt logic 527 flows into a set of registers 529, 531. The register 531 also receives input from the DS3 deframing logic 525. The bits stored in register 531 loop back into the per-channel state memory511. The bits stored in the register 529 flow back into the sync hunt per-alignment memory 513. Output from the DS3 deframing logic 525 is also stored in a register 533 before flowing to the DS2 deframer 322 (as shown in Figure 3).

Please replace paragraph number [00057] with the following amended paragraph:

[00057] Figure 6 is a diagram of the DS2 deframer 322 of Figure 3 according to one embodiment of the invention. The data bit stream 406, the validity bit stream 408 and the subchannel bit stream 410 flow from the register ~~531~~ 533 of the DS3 deframer to the DS2 deframer 322. The bits stored in a register 605 are from the subchannel bit stream 410 and the data bit stream 406. The bits stored in a register 603 are from the validity bit stream 408 ~~of Figure C~~. A dashed line 635 indicates a first pipe stage of the DS2 deframer 322. In the first pipe stage, bits flow from the register 603 to the register 607 and from the register 605 to a register 609. In addition, the bits from the register 605 flow through a per-channel state memory 623 and into a register 611. A dashed line 637 indicates a second pipe stage of the DS2 deframer 322. In the second pipe stage, bits stored in the registers 607, 609 and 611 flow into registers 613, 615 and 617 respectively. The bits from the register 611 also flow through a sync hunt per-alignment memory 621 and into a register 619. A third dashed line 639 indicates a third pipe stage for the DS2 deframer 322. The bits stored in the registers 613, 615, 617 and 619 flow into a DS2 deframing logic 625 and a DS2 sync hunt logic 627. After being processed by the DS2 sync hunt logic 627, bits are stored in a register 633 before flowing back into the sync hunt per-alignment memory 621. Output from both the DS2 deframing logic 625 and the DS2 sync hunt logic 627 is stored in a register 631. From the register 631, bits flow back into the per-channel

state memory 623. Output from the DS2 deframing logic 625 also flows into a register 629 before continuing on to the DS1 deframer 324.

A13

Please replace paragraph number [00060] with the following amended paragraph:

[00060] In one embodiment of the invention, the addressing pointers for the external ~~sync hunt~~ memory unit 323 (shown in Figure 3) are stored in the FIFO core 721, 735. Placing the addressing pointers in the FIFO core 721, 735 reduces the complexity of resetting and controlling the addressing pointers. In another embodiment of the invention, the addressing pointers are stored in the memory controller 321. In one embodiment of the invention, which stores the addressing pointer in the memory controller 321, a register array for each deframer slice is placed in a larger register array that is placed in the memory controller 321. Such a design provides the benefit of reducing the hardware necessary for implementing the deframer.

A14

Please replace paragraph number [00061] with the following amended paragraph:

[00061] As shown by Figures 5-7, each deframer performs both sync hunting and deframing. Sync hunting is performed by the sync hunt logics 527, 627, and 727. After synchronization, a bit stream is deframed by the corresponding one of the deframing logics 525, 625, 725 while the sync hunt logic continues to monitor sync. If a channel gets out of sync, sync hunt for that channel is restarted.

A15

Please replace paragraph number [00063] with the following amended paragraph:

[00063] Synchronizing a bit stream (sync hunting) comprises searching for a bit pattern formed by an alignment signal. For example, a DS3 frame includes seven subframes. Each subframe comprises eight 85 bit blocks. The first bit of each block is an overhead bit which includes bits of the alignment signal. For a DS3 signal, the alignment signal includes F-bits and M-bits. The F-bits or framing bits form a bit pattern "1001" in each subframe at blocks two, four, six, and eight. Each F-bit is separated by 170 bits. The M-bits or multiframing bits form a bit pattern "010". The M-bits occur in the first block of the fifth, sixth, and seventh subframe. It should be understood that the invention is not limited to these bit patterns. In another embodiment of the invention, the logic searches for different bit patterns to synchronize a bit stream or signal. The sync hunting logic 527, 627 maintains multiple per-alignment state machines to be described. The sync hunt logic performs sync hunting concurrently for multiple per-alignment state machines using a single bit. The logic determines if the bit matches the F-bit pattern for one per-alignment state machine and the M-bit pattern for a different per-alignment state machine. The sync hunting is described in more detail with reference to Figures ~~G-M~~8-12.

Please replace paragraph number [00064] with the following amended paragraph:

[00064] Figures 8A-8B are flow charts for DS3 sync hunting performed by the DS3 sync hunt logic 527 of Figure 5 according to one embodiment of the invention. Figure 8A is a flow chart for DS3 sync hunting according to one embodiment of the invention. As indicated in Figure 5, bits are used from the registers 517, 519, 521 and 523. If a bit stored in the register 517 indicates invalidity, then a corresponding signal bit stored in the register 519 is not processed by the following logic. The term signal bit is used to distinguish data bits of the data bit stream from stuffing bits added to the data bit stream by the receiving network element. The signal bits (data bits) can be categorized as payload bits or overhead bits. Although a signal bit may be a payload bit from the perspective of the DS3 deframer, it may be an overhead bit from the perspective of the DS2 or DS1 deframer. The following logic is performed for each subchannel or side.

A17

Please replace paragraph number [00070] with the following amended paragraph:

[00070] At block 823, it is determined if the bit received at block 819 is the next expected F-bit for a per-alignment state machine *X*. If the received bit is the next expected F-bit for the per-alignment state machine *X*, then at block 831 it is determined if *X=N-1*. If at block 823 it is determined that the bit is not the next expected F-bit for the per-alignment state machine *X*, then at block 827 the sync hunt state machine for the per-alignment state machine *X* is set to

A18

indicate failure. From block 827 control flows to block 831. If *X* does not equal *N-1*, then at block 829 *X* is incremented. From block 829 control flows back to the block 819. If it is determined at block 831 that *X* equals *N-1*, then at block 833 it is determined if all per alignment state machines have failed or a time out has occurred. If all of the per-alignment state machines have failed or a timeout has occurred, then at block 835 the DS3 sync hunting restarts. In an alternative embodiment of the invention, a timeout forces the sync hunt logic to select one of the per-alignment state machines which have not failed. If it is determined at block 833 that all of the per-alignment state machines have not failed and a timeout had not occurred, then at block 837 it is determined if only one per-alignment state machine remains valid. If it is determined at block 837 that more than one per-alignment state machine still remains valid, then control flows to block 817. If only one per-alignment state machine remains valid, then it is determined if the per-alignment state machine indicates a state of "MAINTAIN_010" at block 838. If it is determined that the per-alignment state machine ~~does~~ indicates "MAINTAIN_010", then the DS3 framing pattern has been detected and at block 839 DS2 deframing begins. If it is determined at block 838 that the per-alignment state machine does not indicate "MAINTAIN_010", then control flows to block 817.

Please replace paragraph number [00075] with the following amended paragraph:

[00075] Figure 10 is a diagram illustrating organization of the per-alignment state machines in the sync hunt per-alignment memory 513 of Figure 5 according to one embodiment of the invention. In Figure ~~K~~10, the per-alignment state machines are arranged as two columns of 85 per-alignment state machines. This organization of the per-alignment state machines allows the use of a single port register array instead of a dual port register array. This organization also allows the sync hunt logic to accomplish 2 tasks concurrently: both the task of verifying subframe alignment with F-bit patterns for a per-alignment state machine *X* and the task of verifying framing bit ~~patters~~ patterns for a per-alignment state machine (*X*+85) MOD 170. In one embodiment, each per-alignment state machine is 7 bits wide. In another embodiment, each per-alignment state machine is wider.

A19

Please replace paragraph number [00078] with the following amended paragraph:

[00078] Figure 12 illustrates an example of storing bits in DS2 per-alignment state machines as potential alignment bits according to one embodiment of the invention. In Figure ~~L~~12, a bit stream 1201 is received. In this example, there are 147 per-alignment state machines, but only six per-alignment state machines are shown. Bits 0, 1, and 2 of the bits stream 1201 are stored as P0 in per-alignment state machines 1203, 1205, and 1207 respectively. Bits 144, 145

A20

and 146 are stored as P0 in per-alignment state machines 1209, 1211, and 1213 respectively. We now return to Figure 11A.

Please replace paragraph number [00098] with the following amended paragraph:

[00098] Figure 18 is a diagram illustrating the organization of per-alignment state machines in the memory unit ~~321~~ 323 of Figure 3 according to one embodiment of the invention. Although the sync hunt logic 727 of Figure 7 only uses 193 per-alignment state machines for DS1 super frame sync hunting, the memory unit 321 of Figure 3 is of a size sufficient to store 770 of the 772 per-alignment state machines for DS1 extended super frame sync hunting. In Figure 18, the DS1 per-alignment state machines are organized as 18480 rows of 7 per-alignment state machines. (18480 is the product of 6 DS3 pairs * 28 DS1 subchannels *110 rows of 7 per-alignment state machines). Each of the per-alignment state machines is 4 bits wide. The two per-alignment state machines that are not stored in the memory unit 321 of Figure 3 are located on chip with the DS1 deframing unit 209 of Figure 2. Since the external memory unit in this example is 28 bits wide, a total of 110 + 2/7 memory lines are needed for sync hunting DS1 extended super frames. Storing the 2/7 memory line in on-chip memory makes the memory organization and bandwidth supplied by the memory controller uniform. In another embodiment of the invention, the memory unit is expanded to accommodate the 2/7 memory line. In such an

embodiment, the depth of the read/write FIFOs is increased to accommodate a periodic dip in memory bandwidth supplied by the memory controller.